Hierarchical Classification for Live Fish Recognition

Hierarchical Classification for Live Fish Recognition

Phoenix X. Huang
http://homepages.inf.ed.ac.uk/s1064211/
Bastiaan J. Boom
http://homepages.inf.ed.ac.uk/bboom/
Robert B. Fisher
http://homepages.inf.ed.ac.uk/rbf/

School of Informatics
The University of Edinburgh
10 Crichton St
Edinburgh EH8 9AB

Huang, P. X., Boom, B., & Fisher, B. (2012). Hierarchical Classification for Live Fish Recognition. In de Campos , T. (Ed.), British Machine Vision Conference 2012 Student Workshop.

Abstract

Live fish recognition in the open sea is a challenging multi-class classification task. We propose a hierarchical classification approach to recognize live fish from underwater videos. However, the hierarchical method accumulates misclassified samples into deeper layers and these accumulated errors reduce the average accuracy. We propose a set of heuristics to help construct more accurate hierarchical trees and, therefore, control the error accumulation. We create an automatically generated tree based on these heuristics and compare it to a baseline tree on a live fish image dataset. The proposed hierarchical classification method achieves about 4% better accuracy compared to state-of-the-art techniques.

1 Introduction

Live fish recognition in the open sea is fundamentally challenging. In such environments, fish can move freely and illumination levels change frequently. As a result, this task remains an outstanding research problem [8, 12, 14, 17]. Prior research is mainly restricted to constrained environments (*e.g.*, fish tanks [8], conveyor belts [16]). In contrast, this paper investigates novel techniques to perform effective live fish recognition in an unrestricted natural environment.

The fish recognition task is generally a multi-class classification problem, which has become an important and interesting research area since the influence of machine learning theory. Over the last decade, SVM [2] has shown impressive accuracy on the multi-class classification task. SVM is originally designed for a binary classification task. Therefore, to enable multi-class classification, several mechanisms, such as one-vs-one and one-vs-rest, have been developed. This kind of multi-class classifier could be considered as a flat classifier which classifies all classes at the same time [1]. A shortcoming of the flat classifier is that it uses the same features to classify all classes without recognizing that specific classes can be better classified by some customized features. To overcome the problem, one possible solution is to integrate a domain knowledge database with the flat classifier and construct a tree to organize all classes hierarchically [4]. This strategy is called hierarchical classification which inherits from the divide and conquer tactic. Essentially, it uses a hierarchical classification procedure where a customized classifier is trained with specifical features at each level [5]. This method is popular in document and image categorization. Mathis [9] organizes documents hierarchically by making use of the correlations between topical subjects. Deng *et.al.* [3] introduced a new dataset called ImageNet where a large scale hierarchical ontology of images are constructed based on the WordNet knowledge. However, these approaches use pre-defined hierarchical structures without considering how to construct a more accurate tree based on given classes. As a result, a possible solution is to consult to the prior knowledge of the fish taxonomy system which has the similar tree structure. The taxonomy ontology is an academic subject which aims to construct a scientific methodology to systematize animals into their hierarchical categories. Taxon, as the leaf node of the whole tree, is the basement of taxonomy knowledge. For each taxon in the taxonomic tree, there is a top-to-bottom description to identify its hierarchical information which contains several concepts, known as Kingdom, Phylum, Class, Order, Family, Genus, Species. Furthermore, the taxonomy methodology is based on the synapomorphies characteristic from extant of which the taxon is monophyletic, and it indicates the distinction between species, *e.g.* the presence or absence of components (anal-fin, nasal, infraorbitals), specific number (six dorsal-fin spines, two spiny dorsal-fins), particular shape (second dorsal-fin spine long, thick caniniform teeth), *etc.* Lampert *et.al* introduce the attribute-based classification systems especially tailored for the animal classification [7]. We use the taxonomy knowledge to help construct a baseline tree of 10 common fish species (figure 2, left)).

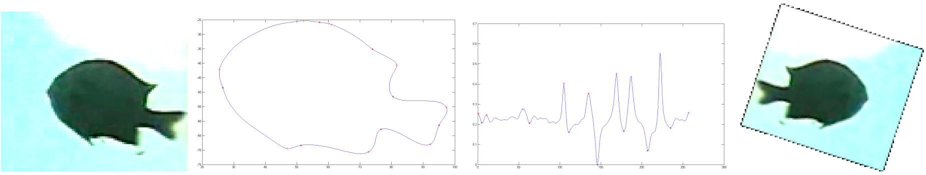
The feature extraction procedure with fish orientation algorithm is introduced in section 2. We discuss the hierarchical classification approach in section 3, and propose a set of heuristics to help construct a better hierarchical tree. In section 4, we evaluate these methods based on a fish image set from the Fish4Knowledge project [11]. We analyze the results of two hierarchical approaches, a baseline tree and an automatically generated tree, and compare them to a flat classifier. Results show that the automatically generated tree has the best performance.

2 Feature extraction

Some pre-processing procedures are undertaken to improve the recognition rate. Firstly, the Grabcut algorithm [13] is employed to segment fish from the background, and produces a binary mask. Secondly, we propose a streamline hypothesis, which uses the assumption that the tail has a abrupt shape because fish need a more frictional tail (caudal fin) to swim and help them keep balance. In order to find the tail side, we smooth the fish boundary with a Gaussian filter to eliminate some noise, and then calculate the curvature of each boundary pixel as following [6, 10]:

$$\kappa(u,\sigma) = \frac{X_u(u,\sigma)Y_{uu}(u,\sigma) - X_{uu}(u,\sigma)Y_u(u,\sigma)}{(X_u(u,\sigma)^2 + Y_u(u,\sigma)^2))^{\frac{3}{2}}} \tag{1}$$

where $X_u(u,\sigma)/X_{uu}(u,\sigma)$ and $Y_u(u,\sigma)/Y_{uu}(u,\sigma)$ are the first and the second derivative of $X(u,\sigma)$ and $Y(u,\sigma)$, respectively; $X(u,\sigma)$ and $Y(u,\sigma)$ are the convolution result of 1-D Gaussian kernel function $g(u,\sigma)$ with fish boundary coordinates $x(u)$ and $y(u)$. However, the pixel curvature is sensitive to local corners and we normalize it using the logarithm



(a) . (b) . (c) . (d) .

Figure 1: Fish orientation demonstration: (a) original fish image; (b) fish boundary after gaussian filter; (c) curvature along fish boundary; (d) oriented fish image.

function:

$$\kappa_{normalize} = \begin{cases} log(\kappa) & \text{if } \kappa \geq 1 \\ -log(2-\kappa) & \text{if } \kappa < 1 \end{cases} \tag{2}$$

The fish boundary coordinates are weighted by their local curvature and the vector from the center of mask to the curvature weighed center estimates the tail orientation. A typical fish orientation procedure is illustrated in Figure 1. The fish orientation method achieves 95% accuracy using 1000 manually labeled fish images. Finally, every fish image is divided into four parts (head/tail/top/bottom) according to the relative positions from the fish center.

This method has achieved a stable accuracy (95%) when finding the tail side in 1000 hand-labeled images. This curvature orientation method selects the relative curvature center which is invariant to the contour scale change. After this, 66 types of feature are extracted. These features are a combination of color, shape and texture properties in different parts of the fish such as tail/head/top/bottom, as well as the whole fish. We use normalized color histogram in the Red&Green channel and the Hue component in HSV color space. These color features are normalized to minimize the affection of illumination changes. We recompute the range of every bin according to the average distribution over all samples and map them into a 11-bin histogram to take full advantage of all bins, as shown below:

$$\widetilde{B_i} = \sum_{j=a_i}^{a_{i+1}} B_j \qquad s.t. \qquad a_i = min\{X \in \mathbb{N}^+ \| \Sigma_{j=1}^{X} \overline{B}_j \geq \frac{i}{11}\} \tag{3}$$

where $B_j, j \in \{1,...,50\}$ is the original color histogram bin, $\overline{B}_j, j \in \{1,...,50\}$ is the averaged histogram over all samples and $\widetilde{B}_i, i \in \{1,...,11\}$ is the recomputed bin.

In order to describe the fish texture, we calculate the co-occurrence matrix, fourier descriptor and gabor filter. The grey level co-occurrence matrices describe the co-occurrence frequency of two grey scale pixels at a given distance d [15]:

$$C_{\Delta u,\Delta v}(i,j) = \sum_{p=1}^{n}\sum_{q=1}^{m} \begin{cases} 1 & \text{if } I(p,q) = i \text{ and } I(p+\Delta u, q+\Delta v) = j \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

The frequency is calculated for several orientations λ. We compute Contrast, Correlation, Energy, Entropy, Homogeneity, Variance, Inverse Difference Moment, Cluster Shade, Cluster Prominence, Max Probability, Auto correlation, Dissimilarity. These 12 features are useful as they are the first selected features by the feature selection procedure.

Histogram of oriented gradients and moment Invariants, as well as Affine Moment Invariants, are employed as the shape features. Furthermore, some specific features like tail/head area ratio, tail/body area ratio, *etc.* are also included. All features are normalized by subtracting the mean and divided by the standard deviation (z-score normalized).

3 Hierarchical classification

Hierarchical classification has several noticeable advantages. Firstly, it divides all classes into certain subsets and leaves similar classes for a later stage. This strategy balances the load of any single node. Secondly, unlike the flat classifier choosing a feature set based on the average accuracy over all classes, the hierarchical method applies a customized set of features to classify specific classes. As a result, it achieves better performance on similar classes [5]. Thirdly, the hierarchical solution exploits the correlations between classes and finds the similar groupings. This is especially useful with a large number of categories [4].

Nonetheless, the hierarchical method has a critical disadvantage called error accumulation. Each level of the hierarchical tree may have some classification errors. These errors are accumulated into deeper layers and reduce the average accuracy of the final result. To solve this problem, we propose a set of rules to help construct a better hierarchical tree and control the error accumulation.

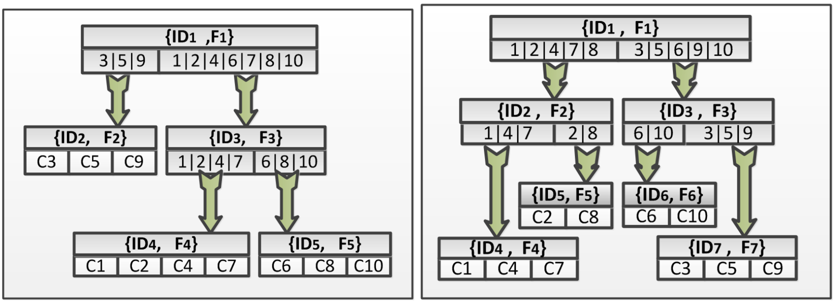
3.1 Hierarchical approach

Given a set of samples $\{x_i\}_{i=1}^n$, the feature vector $f_i = \{f_{i,1}, ..., f_{i,m}\}$ denotes the m feature values for sample x_i. Let $\{y_i\}_{i=1}^n$ indicate the class label of x_i, and $y_i \in \{1, ..., c\}$ where c is the number of classes. Our aim is to construct a classifier h which uses the feature f_i as input to predict the class label $\tilde{y}_i = h(f_i)$ that maximizes the Average Per-Class Accuracy 9.

A hierarchical classifier approach h_{hier} is designed as a structured node set. Fundamentally, a node is defined as a triple: $\text{Node}_t = \{\text{ID}_t, \tilde{\text{F}}_t, \hat{\text{C}}_t\}$, where ID_t is a unique node number t, $\tilde{\text{F}}_t \subset \{f_1, ..., f_m\}$ is a feature subset chosen by a feature selection procedure that are found to be effective for classifying $\hat{\text{C}}_t$, which is a subset of classes and their groups. We only consider binary trees so each node has at most two groups. All samples that are classified as the same group will be transmitted into the same child node for later processing. Two examples with 10 classes are demonstrated in Figure 2. The error accumulation of a hierarchical classification is defined as the samples which are misclassified at the swallow layers and are passed to the deep layers along the wrong group. Take the first node ID_1 in Figure 2(a) for example. If a sample of species C_1 is misclassified to the first group (C_3, C_5, C_9), it will be passed to node ID_2 and classified as either C_3, C_5, C_9. This sample is part of the accumulated errors of the hierarchical tree.

3.2 Rules for tree construction

To construct a hierarchical tree, we first aim at finding an optimal split of the given classes and dividing them into several groups. Then, a customized classifier is trained at the child node for similar classes to make it less likely for them to be misclassified. These similar classes can be observed from the confusion matrix of the classification result. A well-designed hierarchical tree can help improve the accuracy of some confusable classes while suppressing the error accumulation. In this paper, we propose two heuristics for how to organize a single classifier and construct a hierarchical tree with higher accuracy.

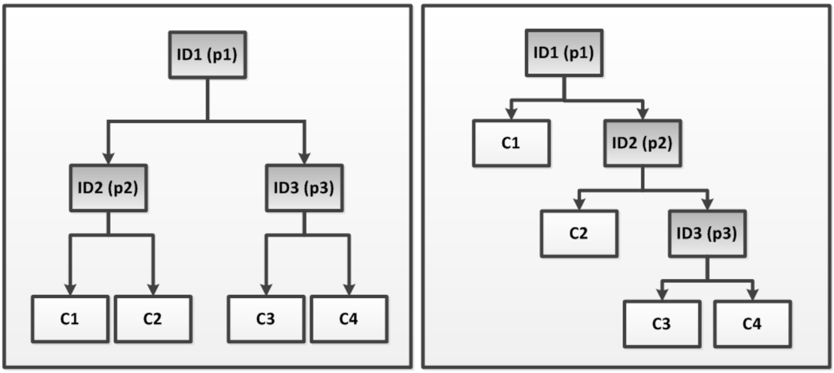


(a) Baseline tree (b) Automatically generated tree

Figure 2: Hierarchical example trees for 10 classes ($C_1, ..., C_{10}$). Fish species name corresponds to Figure 4.

1. Arrange more accurate classifications at a higher level and leave similar classes to deeper layers.

2. Keep the hierarchical tree balanced to minimize the max-depth and control error accumulation.



(a) Balanced tree T_b (b) Unbalanced tree T_u

Figure 3: Examples of hierarchical trees.

Rule 1 recommends how to assign the single classifiers to a hierarchical tree. We consider the balanced tree T_b in Figure 3(a) with sample number n_i. This tree has 4 classes $\{c_1, c_2, c_3, c_4\}$ and each single classifier has a different accuracy $\{p_1, p_2, p_3\}$. The accuracy p is defined as:

$$p = \frac{TruePositive + TrueNegative}{Positive + Negative} \tag{5}$$

The accuracy of first layer is p_1. The average accuracy of the second layer depends on the classification result of the first layer and can be calculated as $p_1 * \frac{1}{2}(p_2 + p_3)$ if we assuming all classes have equal magnitude. The best accuracy is achieved by assigning the most accurate classifier to node ID_1. Generally, the result of a balanced hierarchical tree of

N nodes has depth $log_2 N$ and average accuracy:

$$P_b = \prod_{i=1}^{log_2 N} \widetilde{P_i} = \prod_{i=1}^{log_2 N} \frac{1}{2^{(i-1)}} \sum_{s=2^{(i-1)}}^{2^i - 1} p_s \tag{6}$$

where $\widetilde{P_i}$ is the average accuracy of all nodes in layer i. The hierarchical tree achieves better accuracy if we choose the more accurate classifiers at higher layers which equates to assigning these nodes a higher weight.

Rule 2 is explained by comparing two sample trees: a balanced tree T_b and an unbalanced tree T_u. These examples are shown in Figure 3. Let us assume each class has the same number of samples n_i and each classifier has an equal accuracy p. In T_b, each class is classified with an accuracy p^2, while the average accuracy in T_u is $\frac{1}{4}(p + p^2 + 2p^3)$. We can prove that $P_b > P_u$, for $0.5 < p < 1$. To generalize, a balanced tree of N nodes has average accuracy:

$$P_b = p^{log_2 N} \tag{7}$$

and unbalanced accuracy:

$$P_u = \frac{1}{N}\left(\sum_{i=1}^{N-1} p^i + p^{N-1}\right) \tag{8}$$

for $0.5 < p < 1$, $P_b > P_u$. Thus a more balanced hierarchical tree with $log_2 N$ depth suppresses error accumulation, and achieves better accuracy than an unbalanced tree.

3.3 Algorithm of generating hierarchical tree

The hierarchical tree is based on the two heuristics of the last section: keep hierarchical tree balance and optimize the performance by putting more accurate nodes at the top layers. In the fish recognition task, some species of fish are more similar than others and the similarity is summarized from the confusion matrix. We illustrate the algorithm of generating hierarchical tree below:

```
Input: class C1 to Cn
begin c := [C_1, ..., C_n]
   level := 0
   construct(c, level);
where
proc construct(c, n) ≡
   if n > MAXDEPTH then exit fi;
   comment: find the best binary split of given classes on whole feature set;
   [cLeft, cRight] := ChooseSplit(c);
   comment: The ChooseSplit function splits the class set into equal-size subsets;
   featureSet = FeatureSelection(cLeft, cRight);
   comment: the minimum splitting is set to 3 to limit the max depth;
   if size(cLeft) > 3 then
                     construct(cLeft, n+1)
   fi;
   if size(cRight) > 3 then
                     construct(cRight, n+1)
```

```
    fi;
end
```

An example generated tree is shown in Fig 2, where 10 classes are arranged into 3 layers. The first layer splits all classes into two groups: C1, C2, C4, C7, C8 and C3, C5, C6, C9, C10. Then it chooses the feature subset to maximize the average accuracy of these groups. This procedure keeps on until all groups have less than 4 classes.

4 Experiment with fish recognition

4.1 Hierarchical classification

We introduce the one-vs-one strategy with a voting mechanism to convert the binary SVM into a multi-class classifier [2]. Each class is trained with a set of binary classifier against any other class. Given a feature vector, each binary classifier does a vote according to the classification score. The class with the highest vote is accepted as the prediction result. A sequential forward feature selection algorithm is applied on each classifier to select a particular subset of discriminative features. We do not use "one-vs-rest" strategy since it achieves worse accuracy due to the imbalanced data.

Based on the multi-class classifier, we designed three classifiers (see Figure 2):

1. A flat SVM classifier, which classifies all 10 classes simultaneously, is implemented as the flat baseline classifier.

2. A baseline hierarchical tree is constructed using a top-to-bottom procedure, which establishes the first node by dividing all colorful fish (species 3,5,9) out. This mechanism keeps dividing until no particular similar group is observed. This tree is pre-defined. It reflects the homologous similarity between species.

3. An automatically generated tree is designed by recursively choosing a binary split which has the best accuracy in given classes. We choose binary splitting to keep the tree balanced.

4.2 Results and analysis

Our data is acquired from a live fish dataset with 3179 fish images of the 10 different species shown in Figure 4. This figure shows the fish species name and the numbers of images. As can be seen, the data is very imbalanced where the first two species account for 2564 images. The fish detection and tracking software described in [11] is used to obtain the fish images. The fish species are manually labeled by following instructions from marine biologists.

The experiment is based on the 3179 fish images with a 6-fold cross validation procedure. The training and testing sets are isolated so fish images from the same trajectory sequence are not used during both training and testing. So, there is no mixing training samples from the same sequences in training/testing. Sequential forward feature selection is applied at each node. We then train a customized classifier at each node for specific classes. The time cost of training the best-spliting hierarchical tree takes about 3 hours on a P4 3.2G computer and the running time cost is about 1 second per fish image including feature extraction. Results are listed in Table 1 where the accuracy averaged over all classes is reported, rather than over all fish. This is because of the greatly unbalanced class sizes.

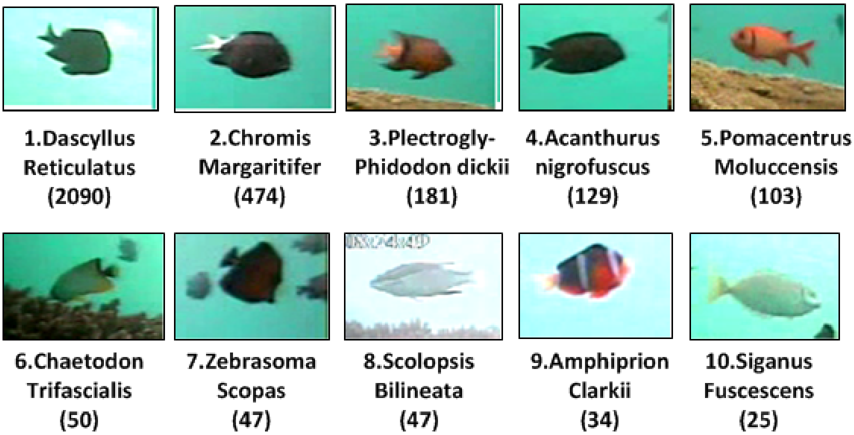


Figure 4: Top 10 species of fish in Taiwan.

The accuracy of a classification system is evaluated as Average Recall (AR). Generally, given True Positive / False Positive / False Negative, the AR is defined as:

$$AR = \frac{1}{c}\sum_{j=1}^{c}\left(\frac{TruePositive_j}{TruePositive_j + FalseNegative_j}\right) \tag{9}$$

where c is the number of classes.

Algorithm	Average accuracy
Flat SVM	86.32 $\pm$5 %
Baseline tree	88.08 $\pm$4 %
Automatically generated tree	90.01 $\pm$4 %

Table 1: Fish recognition result.

We compare the hierarchical classification against the flat SVM classifier (86.32%). The baseline tree is a human-derived model and it recognizes colorful fish with a customized classifier. As a result, the baseline tree achieves a higher average accuracy (88.08%) than the flat SVM but it is worse than the automatically generated hierarchical tree (90.01%) which chooses the best splitting by exhaustively searching all possible combinations while remaining balanced. The search procedure takes several hours and a possible improvement is to integrate the hierarchical method with domain knowledge like taxonomy, which helps organize similar species for later processing, instead of exhaustive searching.

The individual class accuracy is shown in Figure 5. The hierarchical approaches achieve a better accuracy than the flat SVM classifier because they arrange the similar species (1,4,7) into the same group and add fish-tail features to distinguish these species. The baseline tree misclassified some fish from species 10 to species 8. As a result, it achieves a better performance in species 8 but has a lower accuracy in species 10.

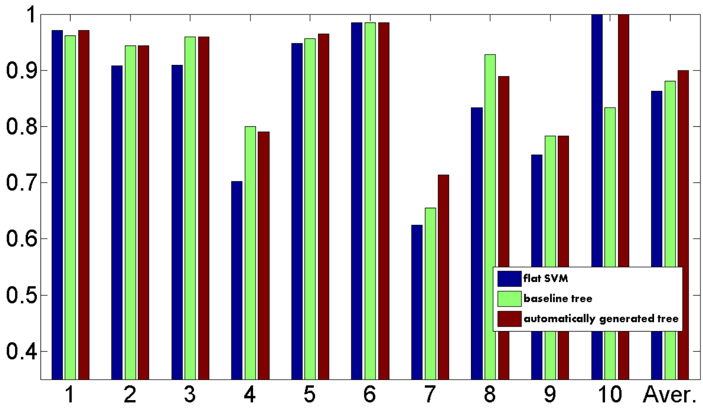


Figure 5: Accuracy of 10 species.

5 Conclusion and future work

In this paper we propose a hierarchical classification approach for live fish recognition. Furthermore, we propose a set of heuristics which are helpful to construct a hierarchical tree. These suggested rules are evaluated with two hierarchical approaches. The experiment is carried out based on a live fish dataset and the automatically generated hierarchical tree achieves about 4% improvement compared to the flat SVM classifier. In the future work, we will investigate the 1-vs-all classifier because the 1-vs-1 classification is not sustainable for large numbers of classes, e.g. 1.5K. Moreover, we will apply the hierarchical classification method to a larger dataset with more fish species.

References

[1] Silla Carlos and Freitas Alex. A survey of hierarchical classification across different application domains. *Data Mining and Knowledge Discovery*, 22(1-2):31–72, 2010.

[2] Chang Chih-Chung and Lin Chih-Jen. LIBSVM: a library for support vector machines. *ACM Trans. Intell. Syst. Technol.*, 2(3):1–27:, 2011. ISSN 2157-6904.

[3] Jia Deng, Wei Dong, R. Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: a large-scale hierarchical image database. CVPR, pages 248 –255, 2009.

[4] Jia Deng, Alexander Berg, Kai Li, and Li Fei-Fei. What does classifying more than 10,000 image categories tell us? In *ECCV*, volume 6315, pages 71–84. 2010. ISBN 978-3-642-15554-3.

[5] Jianping Fan, Yuli Gao, and Hangzai Luo. Hierarchical classification for automatic image annotation. SIGIR, pages 111–118, 2007.

[6] X. C. He and N. H. C. Yung. Curvature scale space corner detector with adaptive threshold and dynamic region of support. In *Pattern Recognition, International Con-*

ference on, volume 2, pages 791–794, Los Alamitos, CA, USA, 2004. IEEE Computer Society.

[7] Christoph Lampert, Hannes Nickisch, and Stefan Harmeling. Learning to detect unseen object classes by between-class attribute transfer. CVPR, 2009.

[8] Dah-Jye Lee, Robert B Schoenberger, Dennis Shiozawa, Xiaoqian Xu, and Pengcheng Zhan. Contour matching for a fish recognition and migration-monitoring system. *Proc. of SPIE*, 5606(1):37–48, 2004.

[9] Charles Mathis. Classification using a hierarchical bayesian approach. volume 4 of *Proc. of ICPR*, pages 103–106, 2002.

[10] F. Mokhtarian and R. Suomela. Robust image corner detection through curvature scale space. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 20(12):1376–1381, 1998. ISSN 0162-8828. doi: 10.1109/34.735812.

[11] G. Nadarajan, Y.-H. Chen-Burger, R.B. Fisher, and C. Spampinato. A flexible system for automated composition of intelligent video analysis. Proc. of ISPA, pages 259–264, 2011.

[12] M. Okamoto, S. Morita, and T. Sato. Fundamental study to estimate fish biomass around coral reef using 3-dimensional underwater video system. In *OCEANS 2000 MTS/IEEE Conference and Exhibition*, volume 2, page 1389–1392, 2000. ISBN 0780365518.

[13] Carsten Rother, Vladimir Kolmogorov, and Andrew Blake. GrabCut: interactive foreground extraction using iterated graph cuts. *ACM Trans. on Graphics (TOG)*, pages 309–314, 2004.

[14] B. P. Ruff, J. A. Marchant, and A. R. Frost. Fish sizing and monitoring using a stereo image analysis system applied to fish farming. *Aquacultural engineering*, 14(2): 155–173, 1995. ISSN 0144-8609.

[15] C. Spampinato, D. Giordano, R. Di Salvo, Y. H.J Chen-Burger, R. B. Fisher, and G. Nadarajan. Automatic fish classification for underwater species behavior understanding. In *Proceedings of the first ACM international workshop on analysis and retrieval of tracked events and motion in imagery streams*, page 45–50, New York, NY, USA, 2010.

[16] N. J. C Strachan. Recognition of fish species by colour and shape. *Image and Vision Computing*, 11:2–10, January 1993. ISSN 0262-8856.

[17] N. J. C. Strachan, P. Nesvadba, and A. R. Allen. Fish species recognition by shape analysis of images. *Pattern Recognition*, 23(5):539–544, 1990. ISSN 0031-3203.